Mixology meets Generative AI

Winter

 # Python # API # LLM

Outline

Introduction

Architecture

Generative AI

Demo

Future Work

Introduction

A personalized cocktail recommendations to avoid potential health risks associated with excessive alcohol consumption.

BarGPT

- Receipe
  - classic?
  - creative?

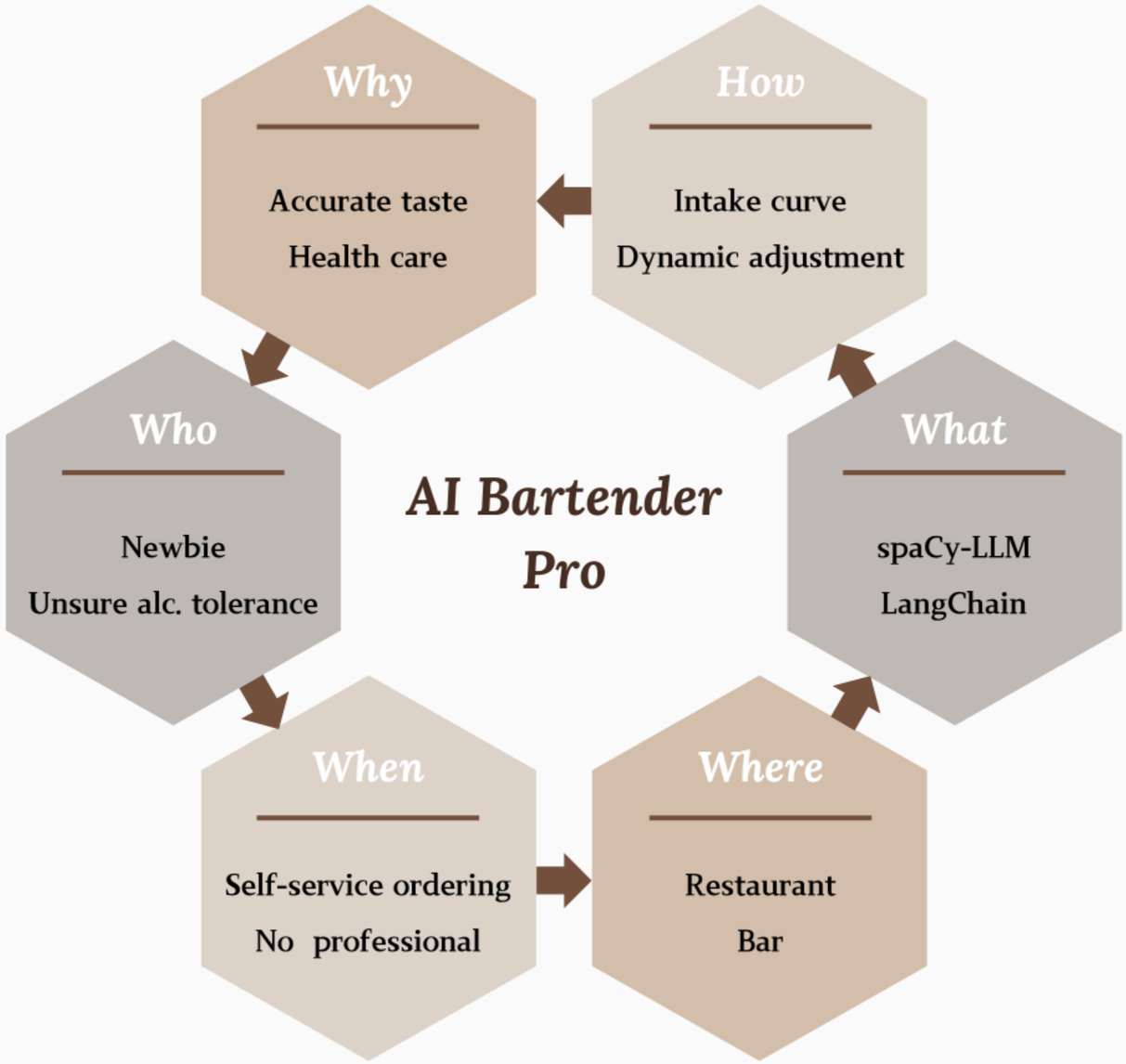
AI Bartender

- Preference
  - base wine?
  - taste?
  - flavor?

AI Bartender Pro

- ***Someone never drank before***
  - find a suitable one
- ***Someone often over-drinking***
  - health care
- ***Be a charming bartender***
  - natural chat
  - emotional understanding
  - wine knowledge

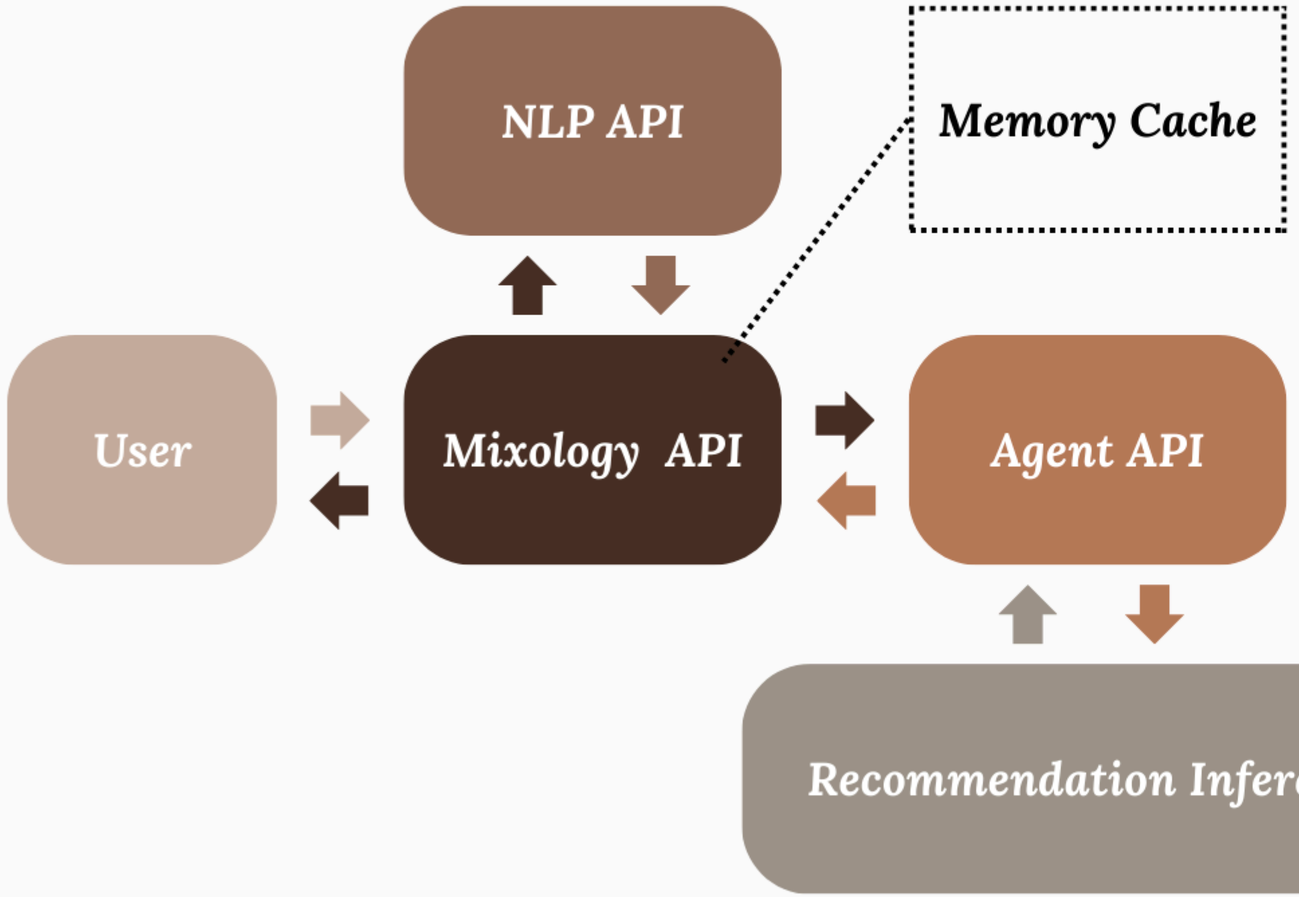
Analysis


Why
Accurate taste
Health care
How
Intake curve
Dynamic adjustment
Who
Newbie
Unsure alc. tolerance
AI Bartender Pro
What
spaCy-LLM
LangChain
When
Self-service ordering
No professional
Where
Restaurant
Bar

Architecture

The architecture on the interaction with user and several APIs.
Simply describe each API's role.

Architecture



Mixology API

The main interface to user, doing task assignment in the whole system.

NLP API

To identify user's meaning as different labels.

Agent API

Take an action according to the labels. such as chat, recommend, etc.

Recommendation Inference

Pre-calculated data from database. Will be updated as the new cocktail items being stored.

Memory Cache

Store the interaction texts.

Generative AI

How open source and generaive AI involved in?

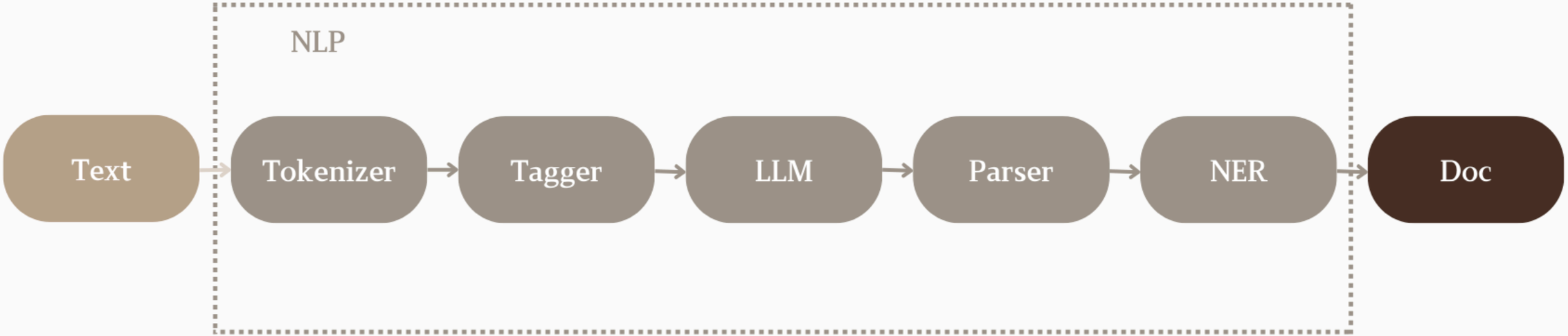
3

spaCy-LLM

“A cutting-edge framework that enhances language modeling capabilities.”

spaCy-LLM Architecture

```
from spacy_llm.util import assemble
```




```
nlp = assemble("classify_question.cfg")
Doc = nlp(Text)
```

spaCy-LLM Chain-of-Thoughts

```
[components.llm.task]
@llm_tasks = "spacy.NER.v3"
labels =  ["greeting", "age_and_gender", "chat", "ask_for_recommend", "choose", "exit"]
description = "User's input can fall into one of the following categories: greeting, age and gender mention, general conversation, request for recommendations, choice making, or indication of exiting the conversation."

[components.llm.task.label_definitions]
greeting = "User's input is a greeting or a social expression of salutation"
age_and_gender = "User's input contains mentions of their age and gender"
chat = "User's input is a general conversation, not involving specific topics such as age, gender, or drinking themes"
ask_for_recommend = "User's input indicates they are requesting recommendations for a cocktail or beverage, typically when seeking suggestions or expressing interest in specific drinks"
choose = "User's input indicates they are making a choice, typically after being provided with recommendations of cocktails or other options"
exit = "User's input indicates they want to exit the conversation"
```

spaCy-LLM In NLP API

```
ent = [(ent.text, ent.label_) for ent in doc.ents]
labels = {}
for i in ent:
        labels[i[1]] = labels.get(i[1], 0) + 1
labels = {name: count * 1.5 if name in {"ask_for_recommend", "choose"} else count for name, count in labels.items()}
predicted = max(labels, key=labels.get) if labels else "unknown"
```

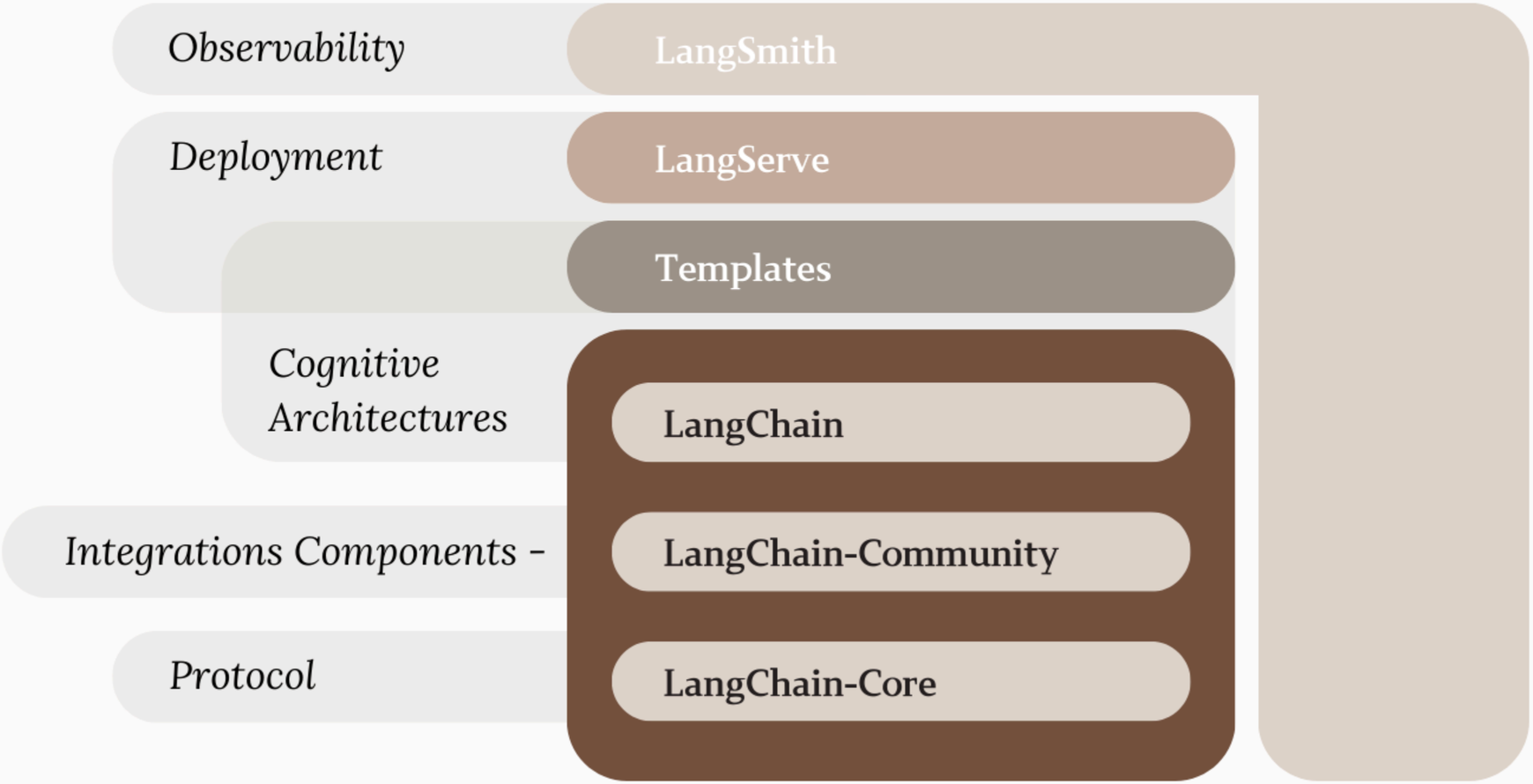
[Weight adjusting]

Input 1: "Yeah. Please recommend me some cocktails." ➡ {"greeting": 1, "ask_for_recommend": 1}
Input 2: "Please recommend me some cocktails." ➡ {"ask_for_recommend": 1}

LangChain

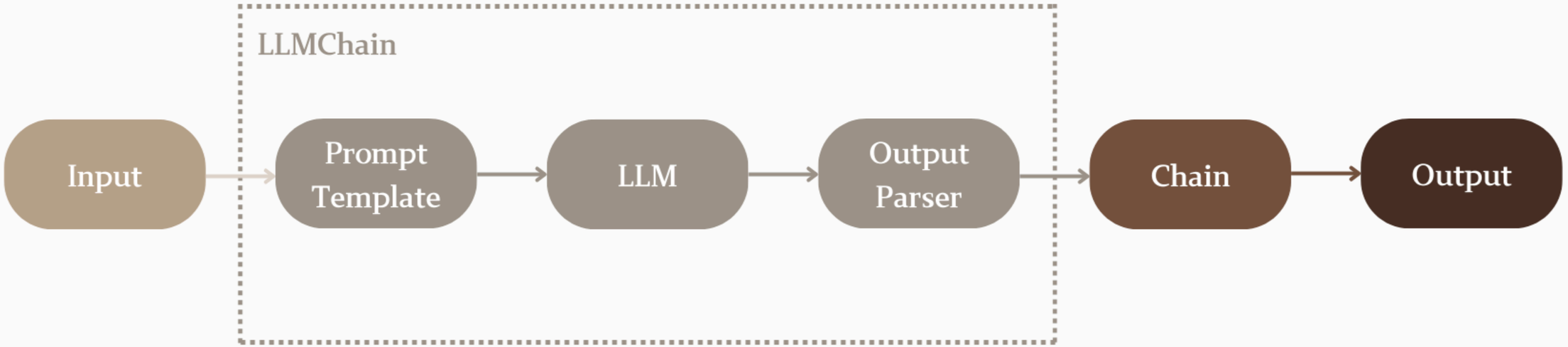
"A framework for developing applications powered by LLMs."

LangChain Architecture



LangChain LLMChain

```
from langchain.chains import LLMChain
```




```
Chain = LLMChain(llm=LLM, prompt=PromptTemplate, output_parser=OutputParser)
Output = Chain.run(Input)
```

LangChain In Parsing Data

Long Island Ice Tea:

15 ml Vodka
15 ml Tequila
15 ml White rum
15 ml Gin
15 ml Cointreau
30 ml Lemon juice
20 ml Simple syrup
Top with Cola

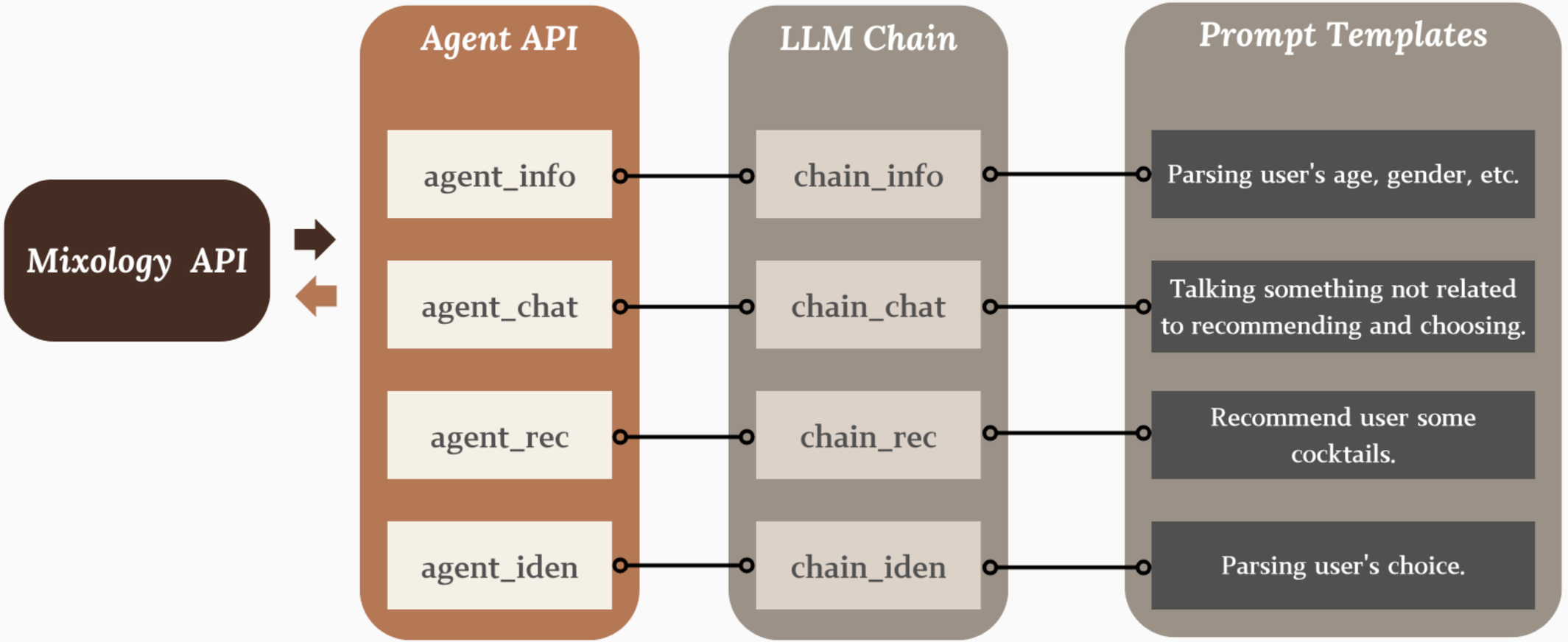
Some Description to identify the dosage, material type, flavor, etc. And make it be a JSON-format output.

LLM Chain

```
{"Long Island Ice Tea":
    ["dosage": "15 ml", "flavor": "spicy",
  "name": "Vodka", "type": "alcohol"], ...
}
```

DB

LangChain In Agent API



Demo

Live demo on the mixology-agent project

Demo

Input info (age, gender, weight)

Ask for recommend and make choice

Need to rest if drink too much (default: 3)

github repo

Future Work

About future direction

5

Future Work

Make the buffer memory to record the whole conversations

Support different needs (flavor, not interested, etc.)

Sincerely welcome for

any critiques or suggestions.